Electronic Version
Stylesheet Version v1.1.1

Description

METHOD AND APPARATUS FOR RLL CODE ENCODING AND DECODING

BACKGROUND OF INVENTION

[0001] 1. Field of the Invention

[0002] The invention relates to an encoding device and a decoding device, and more particularly, to an encoding device and a decoding device for use in a high-density recording system or a high-density data transmission system.

[0003] 2. Description of the Prior Art

[0004] When data is transmitted through a transmission line or recorded onto a recording medium such as a magnetic disc, an optical disc, or a magneto-optic disc, the data is first encoded into a code appropriate for the transmission line or the recording medium. As Kahlman, et al. describe in US Patent 4,477,222 and US Patent 6,496,541, a well known encoding technique is variant of block encoding expressed as Run Length Limited (RLL) (d, k; m, n; r). In

RLL (d, k; m, n; r), m-bit source words are blocked into units each comprising m*i bits. Each unit is converted into a code word having a total code length of n*i bits, and as RLL is a variable length code, i is an integer of at least 1 ranging to a maximum value of r.

[0005] Because of the non-zero switching times of bit transitions, to prevent ambiguity at the receiver between successively received "1"s that could cause a "0" between the "1"s to be misinterpreted as a "1", the parameter d is used to specify the minimum number of "0"s that must appear between two consecutive "1"s. d is referred to as a minimum run of zeros. Additionally, to prevent long runs of "0"s, which would prevent the receiver from doing timing recovery, the parameter k is used to specify the maximum number of "0"s appearing between two consecutive "1"s. k is referred to as a maximum run of zeros.

[0006] Many transmission lines and recording media are not suitable for transmitting or storing an absolute value such as a "0" or a "1" but are well adapted for bit transitions, in other words the act of switching from a "0" to a "1" or vice versa. To accommodate this restriction, the variable length code undergoes an NRZI (Non Return to Zero Inverted) conversion where each "1" of the variable length code is interpreted as a bit inversion while each "0" of the variable length code is interpreted as a non-inversion. The NRZI modulated variable length code is referred to as a recording wave train.

[0007] Let notations Tmin and Tmax denote the minimum and maximum inversion periods of a recording wave train respectively. In order to record the recording wave train at a high recording density, a long minimum inversion period Tmin is preferred, equating to a large minimum run of zeros d. Additionally, from the clock recovery point of view, it is desirable to have a short maximum inversion period Tmax, equating to a small maximum run of zeros k. In order to satisfy these requirements, a variety of encoding techniques have been developed.

[0008] Fig.1 shows a typical conversion table 10 for the variable length RLL (1,7) code. A common encoding technique for optical discs,magnetic discs, or magneto-optic discs is the variable length code RLL (1,7), which can also be expressed as (1, 7; 2, 3; 2).Please note, in Fig.1, the lower case "x" used in the conversion table 10 has the value "1" if the next successive channel bit is a "0" or has the value "0" if the next successive channel bit is a "1".

[0009] The minimum inversion period Tmin, which can be expressed as (d+1)T,for Fig.1 is thus equal to 2T, where T is a bit gap in the recording wave train. The fact that a lot of bit edges are generated at short intervals is helpful for the generation of a clock signal in the receiver. However, as the recording line density is further increased, this minimum run of zeros d adversely affects overall performance. For example, if minimum inversion periods 2T are generated consecutively, the closely spaced waveform in the recording wave train is especially prone to distortion generated by disturbances such as noise. Furthermore, because the dominate error event encountered while reading back data stored in a high density recording system is a 2T interval being misread as a 1T interval, a single error can effectively cause a shift of the front edge of a block of received codewords resulting in a run of bit errors for the entire repeated consecutive minimum runs of zeros d.

[0010] In US Patent 6,496,541, Kahlman et al. describe an RLL(1,7) encoder/decoder that limits the repeated minimum transition runs to a value of six, also expressed as RMTR = 6. The RMTR constraint limits the number of consecutive appearances of minimum runs d to a maximum value of six.

[0011] Fig.2 shows a finite state transition diagram 20 for the bit stream of an RLL(1,7) RMTR=6 constrained system according to the prior art. All compliant bit transitions are indicated on this diagram. As an example of a non-compliant bit stream, suppose the current state is at node A. Node A indicates the immediately preceding bits received were "...1010101010101010" as the only way to reach node A is to pass through this sequence of bits. Node A has only a single path going to node B when a "0" is received. This is because a "0" must be received. If an additional "1" were received, this would mean that the immediately preceding bits were "...1010101010101010", which violates the RMTR=6 constraint. Similarly, due to the maximum run of zeros being specified as k=7, node C has only a single path going to node D when a "1" is received. If an additional "0" were received at node C, this would mean that eight zeros were been received in a row, violating the k=7 constraint.

[0012] Although an RMTR constrained system such as that shown in Fig.2 reduces the resulting run of bit errors for a consecutive run of minimum runs d caused by a 2T interval being misread as a 1T interval, the bit errors that do occur are often uncorrectable. For example if a "1" is received at node A, although the system can detect this as an error, the correct bit stream is not obvious.

[0013] An additional problem with the typical RLL(1,7) encoder/decoders according to the prior art is a maximum run k being misinterpreted as a SYNC signal. Generally speaking, data systems employ consecutive long run length patterns as the SYNC pattern. For example, a typical SYNC pattern is defined as consecutive 9T internals. In an RLL(1,7) based system, the maximum inversion internal is 8T. A timing error (or another error) could occur causing the receiver to misread an 8T interval as a 9T interval. If consecutive errors of this nature occur, a SYNC pattern could be mistakenly received.

SUMMARY OF INVENTION

[0014] It is therefore a primary objective of the claimed invention to provide an encoder and decoder that limits a characteristic of the codeword for each starting bit position in the codeword, to allow correction of errors caused by 2T intervals being misread as 1T internals and to help prevent mistakenly received SYNC patterns.

[0015] According to the claimed invention, a device for encoding a bit stream of data bits of a binary source signal into a stream of data bits of a binary channel signal, the bit stream of the binary source signal being divided into m-bit source words. The device comprises converting means used to convert m-bit source words into codeword having a variable code length with a basic code length of n bits and a total code length of n*i bits, i being an integer of at least 1. The converting means limits a characteristic of the codeword specified for each starting bit position in the code word.

[0016] According to the claimed invention, a device for decoding a bit stream of data bits of a binary channel signal into a stream of data bits of a binary source signal, the bit stream of the binary source signal being divided into m-bit source words. The device comprises converting means used to convert codeword having a variable code length with a basic code length of n bits and a total code length of n*i bits into m-bit source words, i being an integer of at least 1. The converting means limits a characteristic of the codeword specified for each starting bit position in the code word.

[0017] These and other objectives of the claimed invention will no doubt become obvious to those of ordinary skill in the art after reading the following detailed description of the preferred embodiment that is illustrated in the various figures and drawings.

Brief Description of Drawings

[0018] Fig.1 is a conversion table for RLL(1,7; 2, 3; 2) according to the prior art.

[0019] Fig.2 is a finite state transition diagram for an RLL(1,7) RMTR = 6 constrained system according to the prior art.

[0020] Fig.3 is a diagram of the bit positions assigned to the bits in each codeword according to the present invention.

[0021] Fig.4 is a block diagram of an encoder according to present invention.

[0022] Fig.5 is a block diagram of a decoder according to present invention.

[0023] Fig.6 is a first time-variable conversion table for use by the encoder of Fig.3 and the decoder of Fig.4 according to the first embodiment of the present invention.

[0024] Fig.7 is a second time-variable conversion table for use by the encoder of Fig.3 and the decoder of Fig.4 according to the second embodiment of the present invention.

[0025] Fig.8 is a main conversion table for use by the encoder of Fig.3 and the decoder of Fig.4 according to the present invention.

Detailed Description

[0026] Fig.3 shows an example bit stream 30 of a channel signal divided into codeword A, B, C. According to the present invention, each code word is divided into bit positions, labeled $P_1P_2P_3 \ldots P_n$. P1 is the first bit position of the codeword, P2 the second, and so on with Pn being the last bit position. By limiting a characteristic of the codeword specified for each starting bit position, error correction is possible and SYNC pattern detection is more accurate. Because the bit positions represent the bit times, limiting a characteristic of the codeword is also referred to as making a parameter time-variable.

[0027] The following example shows how a time-variable repeated minimum transition run RMTR can be used to assist error correction using an RLL(1,7) based codeword.

[0028] 123123123123123<-- codeword bit positions

[0029] 101010011001010100<-- received code

[0030] 2 2313 2 2 3<-- transitions T

[0031] The received code shown above contains a 1T transition which is illegal in a RLL(1,7) system. Assuming a timing error caused a single bit transition to be received at the wrong time, there are 2 possible ways to correct the received code.

[0032] The first correction is to move the leading-edge of the 1T transition to one bit earlier than the received code, shown below as possible code #1.

[0033] 12312312312312312<-- codeword bit positions

[0034] 10101010100101010O<-- possible code #1

[0035] 2 222 3 223<-- transitions T

[0036] The second correction is to move the trailing edge of the 1T transition to one bit later than the received code, shown below as possible code #2.

[0037] 12312312312312312<-- codeword bit positions

[0038] 10101001010101010O<-- possible code #2

[0039] 2 2322 223<-- transitions T

[0040] In an RMTR =4 constrained system, both possible code #1 and possible code #2 could be correct codewords. By limiting the repeated minimum transition runs RMTR to 3 for the first bit position, 4 for the second bit position, and 3 for the third bit position, expressed as $\mathrm{RMTR}_{\mathrm{VAR}}$=(3,4,3), only possible code #2 is compliant. Possible code #1 is not a correct codeword because possible code #1 has 4 consecutive minimum transition runs 2T starting at bit position 1. Possible code #2 is a compliant codeword because possible code #2 has a 4 consecutive minimum transition runs 2T starting at bit position 2.

[0041] Making the maximum run of zeros k constraint time-variable allows better rejection of mistakenly received SYNC patterns. For example, by limiting the maximum run of zeros k to 6 for the first bit position, 7 for the second bit position, and 6 for the third position, expressed as k_{VAR}=(6,7,6); a consecutive run of 7 zeros (ie. an interval of 8T) can only occur if it starts from the second bit position. Because a typical SYNC pattern is defined as consecutive 9T internals, consecutive timing errors could cause an 8T interval to be misread as a SYNC pattern. In a k_{VAR}=(6,7,6) limited system this can only happen when starting from the second bit position as the encoder will send a maximum of only 6 zeros when starting from the first and third bit positions. In a k_{VAR} constrained system, the difference between the data portion of the codeword and the SYNC pattern is more significant than in a system with only a k constraint.

[0042] Fig.4shows a block diagram of an encoder 40 according to present invention. The encoder 40 comprises an incoming data shift-register 42 connected to an encoding circuit 44, which is connected to an outgoing codeword shift-register 46. Additionally, the encoding circuit 44 includes a main conversion table 48 and a time-variable conversion table 49.

[0043] The encoder 40 converts 2-bit source words to 3-bit codeword and has a maximum code length of r = 5. Serial data bits for a binary source signal are shifted into the incoming data shift-register 42. In Fig.4, D_1D_2 represents the first 2-bit source word, D_3D_4 represents the second 2-bit source word, and so on with D_9D_{10} representing the fifth 2-bit source word. The encoding circuit 44 checks the time variable conversion table 49 and the last code word sent C_{-2} C_{-1} C_0 for a substitution rule to convert the 5 source words to 5 codeword. If a rule exists, the encoding circuit 44 loads the outgoing codeword shift-register (C_1 to C_{15}) with the corresponding 5 codeword. The 5 codeword are shifted out and 5 new source words are shifted into the incoming data shift-register 42.

[0044] If no substitution rule exists for all 5 source words grouped together, the encoding circuit 44 checks the main conversion table 48 for a substitution rule to convert the first 4 source words (D_1 to D_8) to 4 codeword depending on the last code word sent C_{-2} C_{-1} C_0. If a rule exists, the encoding circuit 44 loads the first 4 codeword (C_1 to C_{12}) in the outgoing codeword shift-register 46 with the corresponding 4 codeword. The 4 codeword (C_1 to C_{12}) are shifted out and 4 new data words are shifted into the incoming data shift-register 42.

[0045] Similarly, if no substitution rule exists for the first 4 source words grouped together, the encoding circuit 44 checks the main conversion table 48 for a substitution rule to convert the first 3 source words (D_1 to D_6). Again, if no substitution rule exists, the encoding circuit 44 checks for a rule for the first 2 source words (D_1 to D_4) and finally, if no rule exists, the encoding circuit uses the main conversion table 48 to convert the first source word D_1 D_2 to the first codeword $C_1C_2C_3$. The number of source words for which a match is found is the same number of codeword that are shifted out of the outgoing codeword register 46 and the same number of source words that are shifted into the incoming data shift-register 42.

[0046] Fig.5 shows a block diagram of a decoder 50 according to the present invention. The decoder 50 comprises an incoming codeword shift-register 52 connected to a decoding circuit 54, which is connected to an outgoing data shift-register 56. As in Fig.4, the decoding circuit also includes the main conversion table 48 and the time variable conversion table 49.

[0047] The decoder 50 performs the opposite operation to that of the encoder 40 described in Fig.4. The decoder 50 converts 3-bit codeword to 2-bit data words having a maximum code length of r = 5. Serial codeword bits for a binary codeword signal are shifted into the incoming codeword shift-register 52, which holds a total of 6 codeword. In Fig.5, $C_1C_2C_3$ represent the first 3-bit source word, $C_4C_5C_6$ represent the second 3-bit codeword, and so on with $C_{13}C_{14}C_{15}$ representing the fifth 3-bit codeword. Because the decoder circuit 54 also needs to check the next incoming code word, the sixth codeword, $C_{16}C_{17}C_{18}$, is also included in the incoming codeword shift-register 52. The decoding circuit 54 checks the time variable conversion table 49 depending on the next incoming code word $C_{16}C_{17}C_{18}$ for a substitution rule to convert the 5 codeword (C_1 to C_{15}) into 5 source words (D_1 to D_{10}). If a rule exists and the next incoming code word $C_{16}C_{17}C_{18}$ is not equal to (010), the decoding circuit 54 loads the outgoing data shift-register 56 with the corresponding 5 data words. The 5 data words (D_1 to D_{10}) are shifted out and 5 new codeword are shifted into the incoming codeword shift-register 52. If a rule exists but the next incoming code word $C_{16}C_{17}C_{18}$ is equal to (010), then the decoding circuit 54 checks the main conversion table 48 to convert the first codeword $C_1C_2C_3$ to the first data word $D_1 D_2$. The decoding circuit 54 loads the outgoing data shift-register 56 with the corresponding 1 data word. The 1 data word (D_1D_2) is shifted out and 1 new codeword is shifted into the incoming codeword shift-register 52.

[0048] If no substitution rule exists for all 5 codeword grouped together, the decoding circuit 54 checks the main conversion table 48 for a substitution rule to convert the first 4 codeword (C_1 to C_{12}) to 4 data words depending on the next code word $C_{13}C_{14}C_{15}$. If a rule exists and the next incoming code word $C_{13}C_{14}C_{15}$ is not equal to (010), the decoding circuit 54 loads the first 4 data words in the outgoing data word shift-register 56 with the corresponding 4 data words. The 4 data words (D_1 to D_8) are shifted out and 4 new codeword are shifted into the incoming codeword shift-register 52. If a rule exists but the next incoming code word $C_{13}C_{14}C_{15}$ is equal to (010), then the decoding circuit 54 checks the main conversion table 48 to convert the first codeword $C_1C_2C_3$ to the first data word D_1D_2. The decoding circuit 54 loads the outgoing data shift-register 56 with the corresponding 1 data word. The 1 data word (D_1D_2) is shifted out and 1 new codeword is shifted into the incoming codeword shift-register 52.

[0049] Similarly, if no substitution rule exists for the first 4 codeword (C_1 to C_{12}) grouped together, the decoding circuit 54 checks the main conversion table 48 for a substitution rule to convert the first 3 codeword (C_1 to C_9). If no substitution rule exists, the decoding circuit 54 checks for a rule for the first 2 codeword (C_1 to C_6) and finally, if no rule exists, the decoding circuit 54 uses the main conversion table 48 to convert the first codeword $C_1C_2C_3$ to the first data word D_1 D_2. The number of codeword for which a match is found is the same number of data words that are shifted out of the outgoing data shift-register 56 and the same number of codeword that are shifted into the incoming codeword shift-register 52.

[0050] Fig.6 shows a first time-variable conversion table 60 according to the first embodiment of the present invention. The first embodiment is an RLL variable length code having the parameters: d=1, m=2, n=3, and r=5. The parity of the source words is preserved over the codeword and two characteristics of the RLL code are limited for each starting bit position in the codeword. Specifically, the repeated minimum transition runs RMTR is limited to 3 for the first bit position, 4 for the second bit position, and 3 for the third bit position, expressed as $RMTR_{VAR}$=(3,4,3). Secondly the maximum run of zeros k is limited to 6 for the first bit position, 7 for the second bit position, and 7 for the third position, expressed as k_{VAR}=(6,7,7). The first time-variable conversion table is divided into the following two sections:

[0051] 62:Substitutions for $RMTR_{VAR}$ = (3,4,3) k_{VAR} = (6,7,7) This section provides substitutions needed to ensure that the time variable $RMTR_{VAR}$ constraints and the time variable k_{VAR} constraints are met.

[0052] 64:Substitutions for RMTR = 4 This section provides substitutions to limit the repeated minimum transition runs RMTR to 4.

[0053] When used in the encoder 40, shown in Fig.4, the encoding circuit 44 searches the first time-variable conversion table for a match of the 5 source words (D_1 to D_{10}) and the previous codeword $C_{-2}C_{-1}C_0$. The searches are preformed starting from the top of the table and moving downward through the different sections. Concerning the previous codeword $C_{-2}C_{-1}C_0$, for some values of the source words (D_1 to D_{10}) no previous codeword $C_{-2}C_{-1}C_0$ is specified. This is to be considered as a don't care value for the previous codeword $C_{-2}C_{-1}C_{0}$ and any value is acceptable. The upper case "X" is a don't care value for a single bit, and "Not 000" is any value for the previous codeword $C_{-2}C_{-1}C_{0}$ except for "000".

[0054] When used in the decoder 50, shown in Fig.5, the decoder circuit 54 first searches the first time-variable conversion table 60 for a match of the 5 codeword (C_1 to C_{15}). If no match is found, the decoder circuit 54 checks the main conversion table to search for a shorter match. If a match exists, then the decoder circuit 54 checks the next codeword $C_{16}C_{17}C_{18}$ for the value "010". If a match exists and $C_{16}C_{17}C_{18}$ = "010", the first time variable conversion table 60 is not used and the decoder circuit 34 uses the main conversion table to directly convert the first codeword $C_1C_2C_3$ to data bits D_1D_2. If a match exists and the next codeword $C_{16}C_{17}C_{18}$ is any other value, the decoder circuit 54 searches the first time-variable conversion table 60 for a match of the 5 codeword (C_1 to C_{15}).

[0055] Fig.7 shows a second time-variable conversion table 70 according to the second embodiment of the present invention. The second embodiment is an RLL variable length code having the parameters: d=1, m=2, n=3, and r=5. The parity of the source words is preserved over the codeword and the time-variable parameters are: RMTR_{VAR}=(4,5,4) and k_{VAR}=(6,7,6). The second time-variable conversion table 70 has only one section:

[0056] 72:Substitutions for RMTR_{VAR} = (4,5,4) k_{VAR} = (6,7,6) - This section provides substitutions needed to ensure that the time variable RMTR constraints and the time variable k constraints are met.

[0057] Depending on the chosen embodiment, either the first time-variable conversion table or the second time-variable conversion table can be used in the encoder 40 and decoder 50 according to present invention. The operation of the second time-variable conversion table is the same as that of the first time-variable conversion table described in Fig.6.

[0058] Fig.8 shows a main conversion table 80 according to the present invention. If the encoder circuit 44 or the decoder circuit 54 are unable to find a match in the time-variable conversion table 48, or in the case of the decoder, if the next codeword has a value of "010", the main conversion table 80 is used. The main conversion table 80 is divided into the following six different sections:

[0059] 82:Substitutions for RMTR = 5 - This section provides substitutions to limit the repeated minimum transition runs RMTR to 5.

[0060] 84:Substitutions for RMTR = 6 - This section provides substitutions to limit the repeated minimum transition runs RMTR to 6.

[0061] 86:Substitutions for k = 7 - This section provides substitutions to limit the maximum runs k to 7.

[0062] 88:Substitutions for k = 8 - This section provides substitutions to limit the maximum runs k to 8.

[0063] 90:Substitutions for d = 1 - This section provides substitutions to limit the minimum runs d to 1.

[0064] 92:Basic Substitutions - This section provides the basic substitutions from 2-bit source words to 3-bit codeword and vice versa while preserving the parity of the source words over the codeword.

[0065] When used in the encoder 40, shown in Fig.4, the encoding circuit 44 searches the main conversion table for a match of the m source words (D_1 to D_{2m}) and the previous codeword $C_{-2}C_{-1}C_0$. The searches are preformed starting from the top of the table and moving downward through the different sections. Concerning the previous codeword $C_{-2}C_{-1}C_0$, for some values of the source words (D_1 to D_{2m}) no previous codeword $C_{-2}C_{-1}C_0$ is specified. This is to be considered as a don't care value for the previous codeword $C_{-2}C_{-1}C_0$ and any value is acceptable. The upper case "X" is a don't care value for a single bit, and "Not 000" is any value for the previous codeword $C_{-2}C_{-1}C_0$ except for "000".

[0066] Similar to the first and second time variable conversion tables, the main conversion table is searched starting from the top of the table and moving downward through the different sections. In the case of the decoder 50, shown in Fig.5, the decoder circuit 54 first checks the next codeword for the value "010". For section 82 the next codeword is made up of the bits $C_{16}C_{17}C_{18}$, for sections 84 and 86 the next codeword is made up of the bits $C_{13}C_{14}C_{15.}$ If the next codeword = "010", the decoder uses the Basic Substitutions, section 92, to directly convert the first codeword $C_1C_2C_3$ to data bits D_1D_2. If the next codeword is any other value than "010", the decoder circuit 54 looks for a codeword match in the main conversion table 80. For sections 88, 90, and 92 decoding is independent of the next codeword, and the decoder circuit 54 looks for a codeword match in the main conversion table 80.

[0067] It should also be noted that the code tables are organized such that if the encoder and decoder both search only a subset of the sections, a valid RLL code is produced. For example, if both the encoder and decoder do not use the time-variable conversion table and instead only search downwards in the main conversion table starting at section 84: Substitutions for RMTR = 6, the resulting RLL code will be RLL (1,7; 2,3; 5) with RMTR = 6; if the search starts at section 86: Substitutions for k = 7, the resulting RLL code will be RLL (1,7; 2,3; 5), etc.

[0068] In contrast to the prior art, the present invention encoder and decoder limits a characteristic of the codeword for each starting bit position in the codeword so that errors caused by 2T intervals being misread as 1T intervals can be corrected and mistakenly received SYNC patterns can be better detected.

[0069] The encoder generates codeword preserving the same parity as the source words. Similar to US Patent 4,477,222, this characteristic can be utilized for DC-suppressing purposes to generate DC-free signals.

[0070] Those skilled in the art will readily observe that numerous modifications and alterations of the device may be made while retaining the teachings of the invention. Accordingly, that above disclosure should be construed as limited only by the metes and bounds of the appended claims.